Writing Good Software Engineering Research Papers

Based on the paper

Mary Shaw, **Writing Good Software Engineering Research Papers**

Proceedings of the 25th International Conference on Software Engineering, IEEE Computer Society, 2003, pp. 726-736.

Research Papers

- The basic and most important activities of the research
  - Visible results, quality stamp
  - Means for communications with other researchers

A good research paper should answer a number of questions

- **What, precisely, was your contribution?**
  - What question did you answer?
  - Why should the reader care?
  - What larger question does this address?
- **What is your new result?**
  - What new knowledge have you contributed that the reader can use elsewhere?
  - What previous work (yours or someone else's) do you build on? What do you provide a superior alternative to?
  - How is your result different from and better than this prior work?
  - What, precisely and in detail, is your new result?
- **Why should the reader believe your result?**
  - What standard should be used to evaluate your claim?
  - What concrete evidence shows that your result satisfies your claim?

If you answer these questions clearly, you'll probably communicate your result well.

Maturity of software engineering discipline

- Other fields of science and engineering (physics, medicine…) – well known methods
- Software engineering – still not well developed and understood research/ presentation guidance

1. What, precisely, was your contribution?

- To precisely answer this, proper (research) questions should be stated

- **What kinds of questions do software engineers investigate?**

Table 1. Types of software engineering research questions	
Type of question	**Examples**
Method or means of development	How can we do/create/modify/evolve (or automate doing) X? What is a better way to do/create/modify/evolve X?
Method for analysis or evaluation	How can I evaluate the quality/correctness of X? How do I choose between X and Y?
Design, evaluation, or analysis of a particular instance	How good is Y? What is property X of artifact/method Y? What is a (better) design, implementation, maintenance ,,, How does X compare to Y? What is the current state of X / practice of Y?
Generalization or characterization	Given X, what will Y (necessarily) be? What, exactly, do we mean by X? What is a good formal/empirical model for X? What are the varieties of X, how are they related?
Feasibility study or exploration	Does X even exist, and if so what is it like? Is it possible to accomplish X at all?

Which type of questions dominate?

- Human-Computer Interaction: - many new trends break through
- Software Engineering
  - mostly incremental (improved model, improved technique)

Table 2. Types of research questions represented in ICSE 2002 submissions and acceptances			
Type of question	**Submitted**	**Accepted**	**Ratio Acc/Sub**
Method or means of development	142(48%)	18 (42%)	(13%)
Method for analysis or evaluation	95 (32%)	19 (44%)	(20%)
Design, evaluation, or analysis of a particular instance	43 (14%)	5 (12%)	(12%)
Generalization or characterization	18 (6%)	1 (2%)	(6%)
Feasibility study or exploration	0 (0%)	0 (0 %)	(0%)
TOTAL	298(100.0%)	43 (100.0%)	(14%)

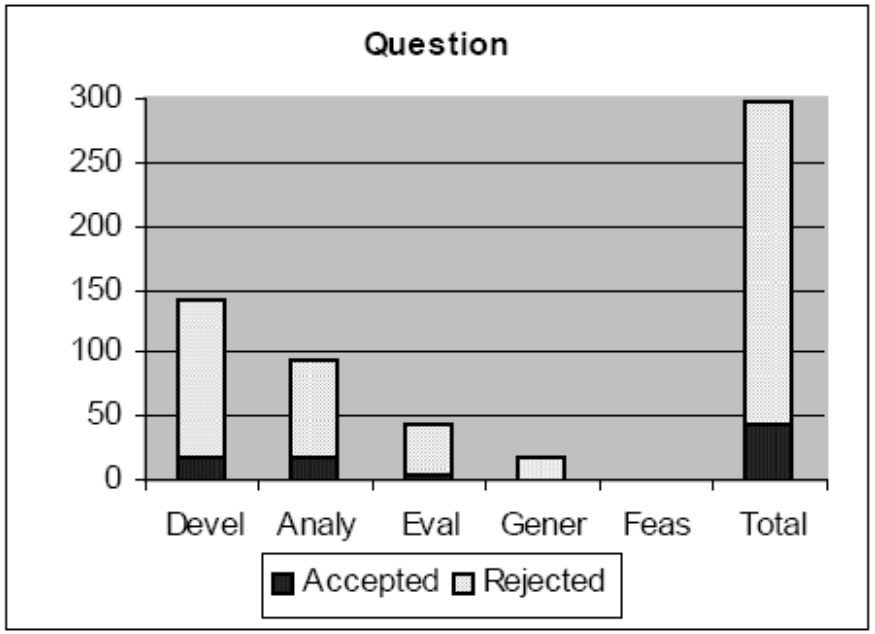


Figure 1. Counts of acceptances and rejections by type of research question

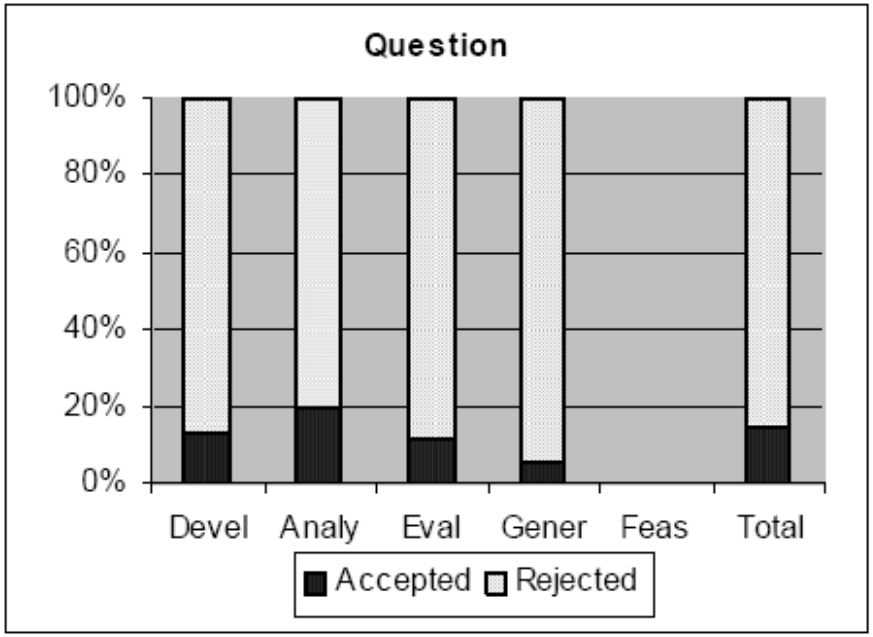


Figure 2. Distribution of acceptances and rejections by type of research question

What do program committees look for?

- The program committee looks for
  - a clear statement of the specific problem you solved
  - *the question about software development you answered*
  - an explanation of how the answer will help solve an important software engineering problem.

You'll devote most of your paper to describing your result, *but you should begin by explaining what question you're answering* and *why the answer matters.*

2. What is your new result?

- Explain precisely
  - *what you have contributed* to the store of software engineering knowledge
  - *how this is useful beyond your own project*.

Table 3. Types of software engineering research results	
Type of result	**Examples**
Procedure or technique	New or better way to do some task, such as design, implementation, maintenance, measurement, evaluation, selection from alternatives; includes techniques for implementation, representation, management, and analysis; a technique should be operational—not advice or guidelines, but a procedure
Qualitative or descriptive model	Structure or taxonomy for a problem area; architectural style, framework, or design pattern; non-formal domain analysis, well-grounded checklists, well-argued informal generalizations, guidance for integrating other results, well-organized interesting observations
Empirical model	Empirical predictive model based on observed data
Analytic model	Structural model that permits formal analysis or automatic manipulation
Tool or notation	Implemented tool that embodies a technique; formal language to support a technique or model (should have a calculus, semantics, or other basis for computing or doing inference)
Specific solution, prototype, answer, or judgment	Solution to application problem that shows application of SE principles – may be design, prototype, or full implementation; careful analysis of a system or its development, result of a specific analysis, evaluation, or comparison
Report	Interesting observations, rules of thumb, but not sufficiently general or systematic to rise to the level of a descriptive model.

Table 4. Types of research results represented in ICSE 2002 submissions and acceptances			
Type of result	**Submitted**	**Accepted**	**Ratio Acc/Sub**
Procedure or technique	152(44%)	28 (51%)	18%
Qualitative or descriptive model	50 (14%)	4 (7%)	8%
Empirical model	4 (1%)	1 (2%)	25%
Analytic model	48 (14%)	7 (13%)	15%
Tool or notation	49 (14%)	10 (18%)	20%
Specific solution, prototype, answer, or judgment	34 (10%)	5 (9%)	15%
Report	11 (3%)	0 (0%)	0%
TOTAL	348(100.0%)	55 (100.0%)	16%

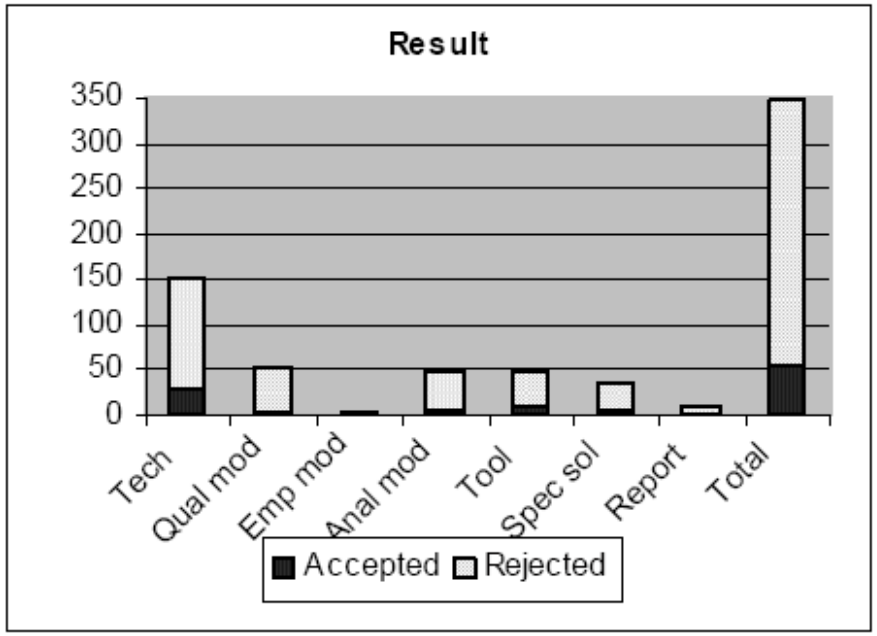


Figure 3. Counts of acceptances and rejections by type of result

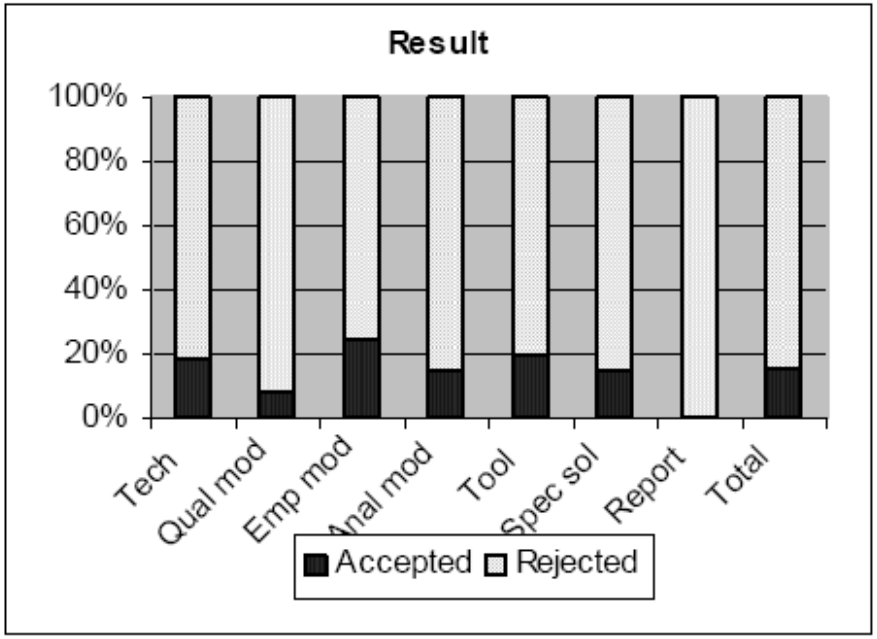


Figure 4. Distribution of acceptances and rejections by type of result

What do program committees look for?

- The program committee looks for
  - interesting, novel, exciting results that significantly enhance our ability
    - to develop and maintain software
    - to know the quality of the software we develop
    - to recognize general principles about software
    - or to analyze properties of software.
- You should explain your result in such a way that someone else could use your ideas.

What do program committees look for? *What's new here?*

Awful	▼	• I completely and generally solved … (unless you actually did!)
Bad	▼	• I worked on galumphing. (or studied, investigated, sought, explored)
Poor	▼	• I worked on improving galumphing. (or contributed to, participated in, helped with)
Good	▲	• I showed the feasibility of composing blitzing with flitzing. • I significantly improved the accuracy of the standard detector. (or proved, demonstrated, created, established, found, developed)
Better	▲	• I automated the production of flitz tables from specifications. • With a novel application of the blivet transform, I achieved a 10% increase in speed and a 15% improvement in coverage over the standard method.

Use verbs that shows Results Not only efforts

Try not. DO, or DO NOT. There is no Try / YoDA


STAR WARS
EPISODE III
REVENGE OF THE SITH
MASTER YODA

What do program committees look for? *What has been done before? How is your work different or better?*

- What existing technology does your research build on?
- What existing technology or prior research does your research provide a superior alternative to?
- What's new here compared to your own previous work?
- What alternatives have other researchers pursued?
- How is your work different or better?

Explain the relation to other work clearly ...

Awful	▼	The galumphing problem has attracted much attention [3,8,10,18,26,32,37]
Bad	▼	Smith [36] and Jones [27] worked on galumphing.
Poor	▼	Smith [36] addressed galumphing by blitzing, whereas Jones [27] took a flitzing approach.
Good	▲	Smith's blitzing approach to galumphing [36] achieved 60% coverage [39]. Jones [27] achieved 80% by flitzing, but only for pointer-free cases [16].
Better	▲	Smith's blitzing approach to galumphing [36] achieved 60% coverage [39]. Jones [27] achieved 80% by flitzing, but only for pointer-free cases [16]. We modified the blitzing approach to use the kernel representation of flitzing and achieved 90% coverage while relaxing the restriction so that only cyclic data structures are prohibited.

What do program committees look for? *What, precisely, is the result?*

- Explain what your result is and how it works. Be concrete and specific. Use examples.
- Example: *system implementation*
- *If the implementation demonstrates an implementation technique*, how does it help the reader use the technique in another setting?
- *If the implementation demonstrates a capability or performance improvement*, what concrete evidence does it offer to support the claim?
- *If the system is itself the result*, in what way is it a contribution to knowledge? Does it, for example, show you can do something that no one has done before

3.
Why should the reader believe your result?

- Show evidence that your result is valid—that it actually helps to solve the problem you set out to solve.

What kinds of validation do software engineers do?

Table 5. Types of software engineering research validation	
Type of validation	Examples
Analysis	I have analyzed my result and find it satisfactory through rigorous analysis, e.g. … For a formal model … rigorous derivation and proof For an empirical model … data on use in controlled situation For a controlled experiment … carefully designed experiment with statistically significant results
Evaluation	Given the stated criteria, my result... For a descriptive model … adequately describes phenomena of interest … For a qualitative model … accounts for the phenomena of interest… For an empirical model … is able to predict … because …, or … generates results that fit actual data … Includes feasibility studies, pilot projects
Experience	My result has been used on real examples by someone other than me, and the evidence of its correctness/usefulness/effectiveness is … For a qualitative model … narrative For an empirical model or tool … data, usually statistical, on practice For a notation or technique … comparison of systems in actual use
Example	Here's an example of how it works on For a technique or procedure …a "slice of life" example based on a real system … For a technique or procedure …a system that I have been developing … For a technique or procedure … a toy example, perhaps motivated by reality The "slice of life" example is most likely to be convincing, especially if accompanied by an explanation of why the simplified example retains the essence of the problem being solved. Toy or textbook examples often fail to provide persuasive validation, (except for standard examples used as model problems by the field).
Persuasion	I thought hard about this, and I believe passionately that ... For a technique … if you do it the following way, then … For a system … a system constructed like this would … For a model … this example shows how my idea works Validation purely by persuasion is rarely sufficient for a research paper. Note, though, that if the original question was about feasibility, a working system, even without analysis, can suffice
Blatant assertion	No serious attempt to evaluate result. This is highly unlikely to be acceptable

Table 6. Types of research validation represented in ICSE 2002 submissions and acceptances			
Type of validation	**Submitted**	**Accepted**	**Ratio Acc/Sub**
Analysis	48 (16%)	11 (26%)	23%
Evaluation	21 (7%)	1 (2%)	5%
Experience	34 (11%)	8 (19%)	24%
Example	82 (27%)	16 (37%)	20%
Some example, can't tell whether it's toy or actual use	6 (2%)	1 (2%)	17%
Persuasion	25 (8%)	0 (0.0%)	0%
No mention of validation in abstract	84 (28%)	6 (14%)	7%
TOTAL	300(100.0%)	43 (100.0%)	14%

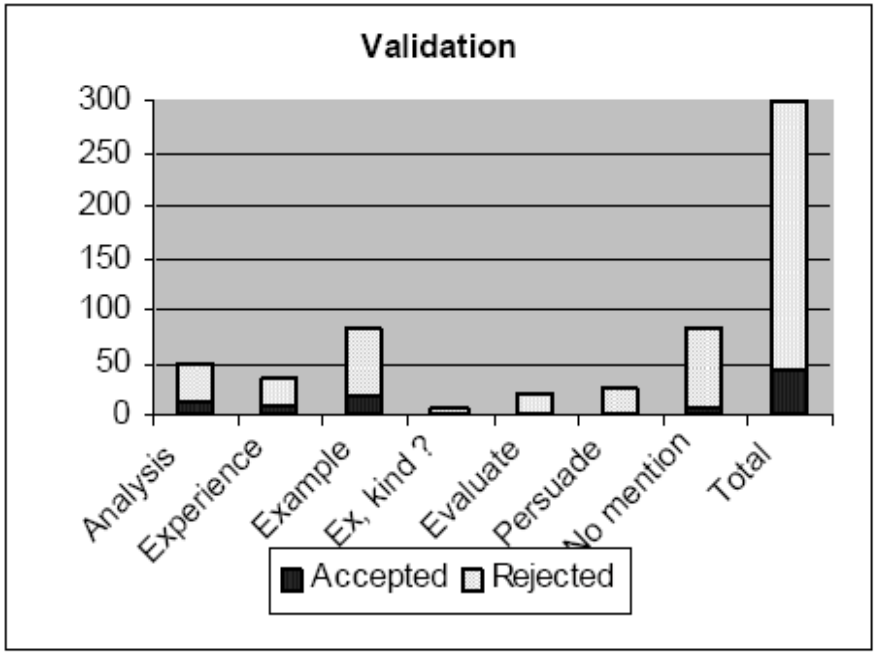


Figure 5. Counts of acceptances and rejections by type of validation

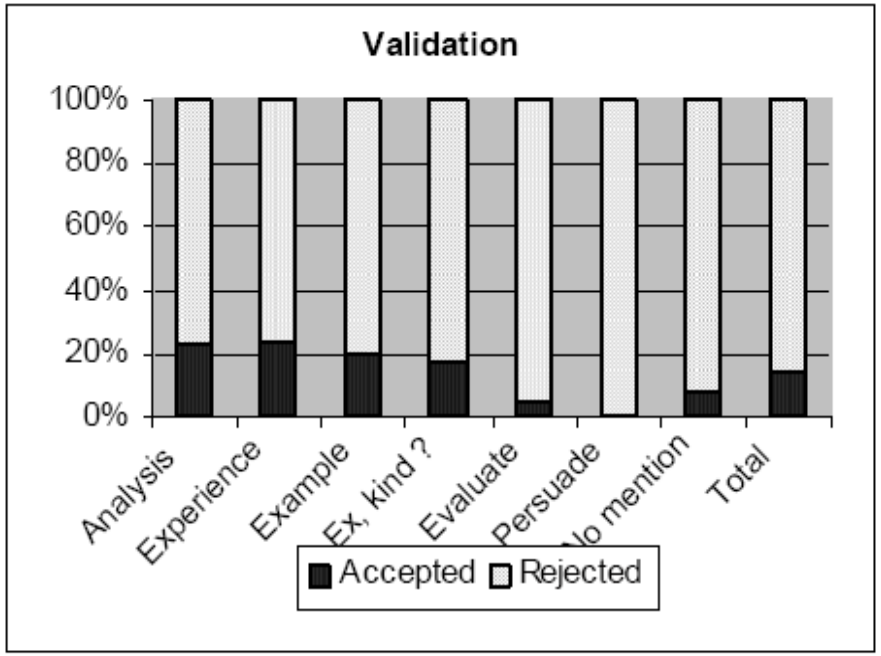


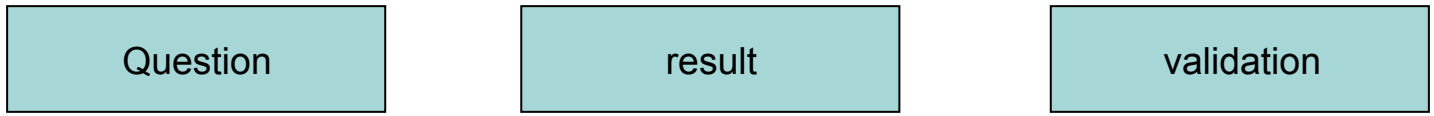
Figure 6. Distribution of acceptances and rejections by type of validation

What do program committees look for? *Why should the reader believe your result?*

- *If you claim to improve on prior art,* compare your result objectively to the prior art.
- *If you used an analysis technique,* follow the rules of that analysis technique.
- *If you offer practical experience as evidence for your result,* establish the effect your research has. If at all possible, compare similar situations with and without your result.
- *If you performed a controlled experiment,* explain the experimental design. What is the hypothesis? What is the treatment? What is being controlled?
- *If you performed an empirical study,* explain what you measured, how you analyzed it, and what you concluded.

4.
How do you combine the elements into a research strategy?

- Not all combinations of a research question, a result, and a validation strategy lead to good research.



Combination question - research - validation

Table 7. Paradigms of ICSE2002 acceptances			
Question	**Result**	**Validation**	#
Devel method	Procedure	Analysis	2
Devel method	Procedure	Experience	3
Devel method	Procedure	Example	3
Devel method	Qual model	Experience	2
Devel method	Analytic model	Experience	2
Devel method	Notation or tool	Experience	1
Analysis method	Procedure	Analysis	5
Analysis method	Procedure	Evaluation	1
Analysis method	Procedure	Experience	2
Analysis method	Procedure	Example	6
Analysis method	Analytic model	Experience	1
Analysis method	Analytic model	Example	2
Analysis method	Tool	Analysis	1
Eval of instance	Specific analysis	Analysis	3
Eval of instance	Specific analysis	Example	2

5.
Does the abstract matter? (YES)

- people judge papers by their abstracts and read the abstract in order to decide whether to read the whole paper.
- It's important for the abstract to tell the story.
- Don't assume, though, that simply adding a sentence about analysis or experience to your abstract is sufficient; the paper must deliver what the abstract promises

5.
Example of an abstract structure:

- Two or three sentences about the current state of the art, identifying a particular problem
- One or two sentences about what this paper contributes to improving the situation
- One or two sentences about the specific result of the paper and the main idea behind it
- A sentence about how the result is demonstrated or defended

Is this presentation a receipt how to succeed?

- Hm?
- Several other conferences offer "how to write a paper"
- advice:
- In 1993, several OOPSLA program committee veterans gave a panel on "How to Get a Paper Accepted at OOPSLA"
- Partridge offers advice on "How to Increase the Chances Your Paper is Accepted at ACM SIGCOMM" [15].
- SIGCHI offers a "Guide to Successful Papers Submission" that includes criteria for evaluation and discussion of common types of CHI results, together with how different evaluation criteria apply for different types of results [13].
- The SIGGRAPH conference program chair wrote a discussion of the selection process, "How to Get Your SIGGRAPH Paper Rejected" [10].
-
- The 2003 SIGGRAPH call for papers [21] has a description of the review process and a frequently-asked questions section with an extensive set of questions on "Getting a Paper Accepted".

Example

- **Challenges of component-based development**
- **Ivica Crnkovic, Magnus Larsson**

- The paper presented at ICSE 2000, as the first paper on the conference
- Selected as one between three papers published in JSS

Challenges of component-based development Abstract

- It is generally understood that building software systems with components has many advantages but the difficulties of this approach should not be ignored. System evolution, maintenance, migration and compatibilities are some of the challenges met with when developing a component-based software system.
- Since most systems evolve over time, components must be maintained or replaced. The evolution of requirements affects not only specific system functions and particular components but also component-based architecture on all levels. Increased complexity is a consequence of different components and systems having different life cycles.
- In component-based systems it is easier to replace part of system with a commercial component. This process is however not straightforward and different factors such as requirements management, marketing issues, etc., must be taken into consideration.
- In this paper we discuss the issues and challenges encountered when developing and using an evolving component-based software system. An industrial control system has been used as a case study.

Motivation
Problem description

Paper Overview:
(Implicit question)
what is the result
validation

Paper outline

1. Introduction
2. **The Case Study**
3. **Different Aspects of Reuse**
4. **Integrating Standard Components**
5. Conclusion

Introduction

- Reuse and an open component-based architecture are the keys to the success of systems with a long lifecycles. Designing a system that supports this approach, requires more effort in the design phase and the time to market might be longer, but in the long run, the reusable architecture will prove profitable.
- …..
- On each level of reuse there are specific demands on the reusable components, on the component management and on the integration process.

- This paper describes important issues related to the development and maintenance of reusable components and as an example uses the ABB Advant industrial process control system.

- In section 2 we give an overview of the Advant system design and the main characteristics of Advant reusable components. Section 3 outlines all the development and maintenance aspects of a component based system which must comply with customer requirements. During evolution of the system new technologies were developed which resulted in the appearance on the market of many components with the same functionality as the proprietary ones. The fact that new components must be incorporated into the existing systems introduces new demands on the system development process. These new issues are discussed in section 4.

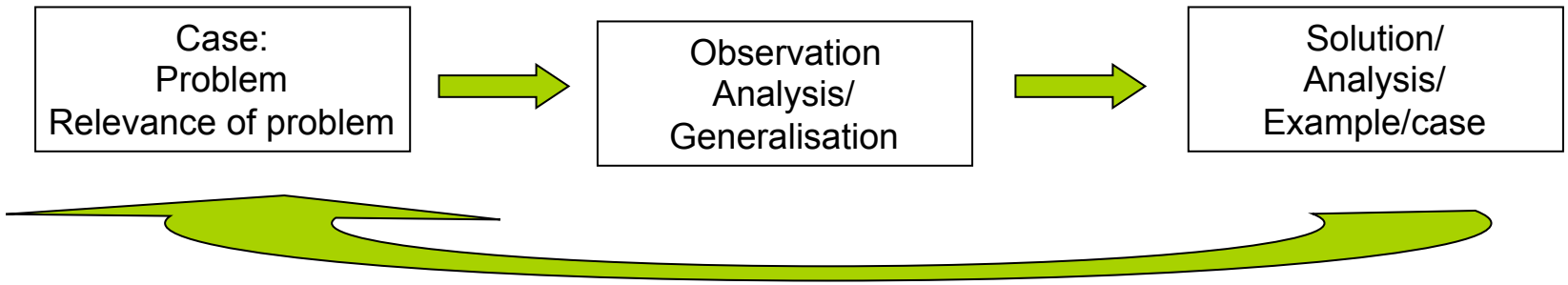
Motivation

Problem description

Paper Overview:
- result

Detailed overview

Story/concept: the pattern



Success factors:

- Realistic situations
- relevant problem
- up/to date problem
- Holistic approach
- Technically sound
- systematical validation trough the case

Weak side of the paper:

- Related work missing
- question(s) not explicitly stated